L4: Entry 8 of 42 File: USPT May 7, 2002

DOCUMENT-IDENTIFIER: US 6385641 B1
**** See image for Certificate of Correction ****
TITLE: Adaptive prefetching for computer network and web browsing with a graphic user interface

Detailed Description Text (107):
Assume that for each user in the system except user k, the current prefetch request rate .lambda..sub.i.sub..sub.2 is known, and .lambda..sub.k.sub..sub.2 =0. We want to determine the prefetch threshold for user k, H.sub.k, such that the cost C decreases if user k prefetches the files with access probabilities greater than or equal to H.sub.k. Similar to the method we used above, we take the derivative of the cost function twice. For the case where ##EQU58##

Detailed Description Text (114):
Up to this point we have explained the algorithms behind the operation of our invention. We have presented a prediction algorithm for web browsing that can be used on both the server and client. It is important to understand how the access probabilities from the server and client should be used, and how this use affects the system performance. Now, we show the experimental results obtained from the UCLA (University of California Los Angeles Computer Science (CS) department web server's log file taken during the period from April 11th to June 5th of 1996. The log consists of more than 300,000 actual user accesses. In the experiments, we concentrated on two representative pages: the CS department home page and the home page for TAs (teaching assistants). More precisely, only when one of these two pages was being viewed, was the prediction algorithm activated and the program used to prefetch files linked to the page. And only when a file linked to one of these two pages was requested, did the program check if it had been prefetched. The CS department home page contains links to research, students, faculty, and other general information home pages. These links are not updated very often. The TA home page has links to all the class home pages, which are updated frequently to include new assignments, handouts, etc. Therefore, simulations on these two pages indicate how well our prediction algorithm works on pages that are revisited frequently by some particular users and on other pages that are generally not revisited frequently. To study how the access probabilities from the server and the client affect the performance of prediction, we tested three cases. 1) Using only the access probabilities from the client site. 2) Using only the access probabilities from the server site. 3) Merging the access probabilities from server and client as designed in our original algorithm. Namely, assume A is the current page; if C.sub.A <5$, then use the access probabilities from the server for the links on page A, otherwise use the access probabilities from the client. A fixed prefetch threshold was used in each experimental run and varied for different runs. All the files with access probability greater than or equal to the prefetch threshold were prefetched. We measured several different system parameters. If the user requested a page, and the request was satisfied by a prefetched copy of the file, we scored the result as a hit. The hit rate is defined as the ratio of the number of hits over the total number of user requests. For a fixed number of user requests, the higher the hit rate, the more time is saved by prefetching. The successful prediction rate is the probability of a prefetched file being eventually used. A high successful prediction rate indicates that less bandwidth is wasted due to prefetching unused files. FIGS. 9-12 show the successful prediction rate and hit rate obtained for the pages linked to the CS home page and the TA home page respectively with prefetch thresholds between 0.01-0.9 for the three cases described above.

Detailed Description Text (134):
Proof: We assume f(q) is continuous. The proof is similar for f(q) discrete. Since f(q) is the rate at which files with access probability q appear in the system, for any h(0<h.ltoreq.1), if all and only those files with access probability greater than or equal to h are prefetched, then the arrival rate of the prefetch requests is ##EQU66##